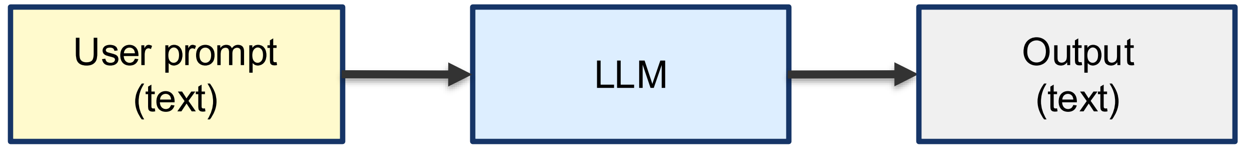
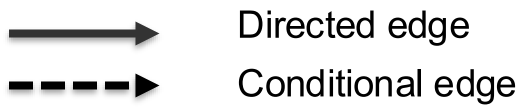
Agentic AI for Scientific Workflows

Murat Keçeli and Thang Pham
Argonne National Laboratory

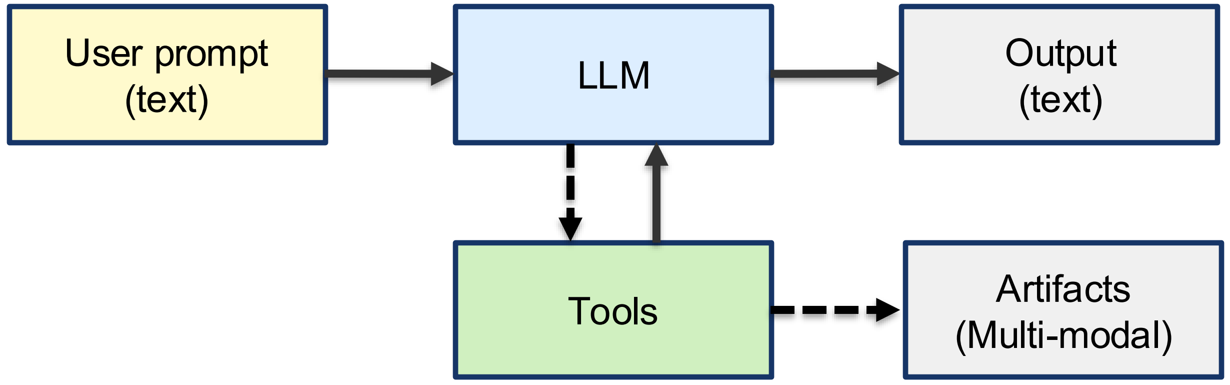
What is Agentic AI?

LLM:





LLM (AI) Agents = LLM + tools



Agentic AI systems can proactively plan, decide, and act. They combine reasoning, tools (APIs, code, simulators), and memory to pursue goals over multiple steps, often coordinating multiple specialized agents for robustness and scale.

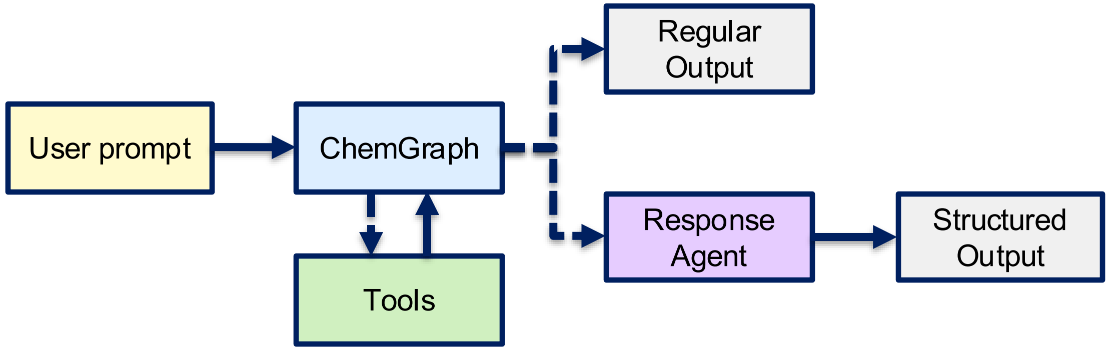
How Agentic AI Helps Scientific Workflows

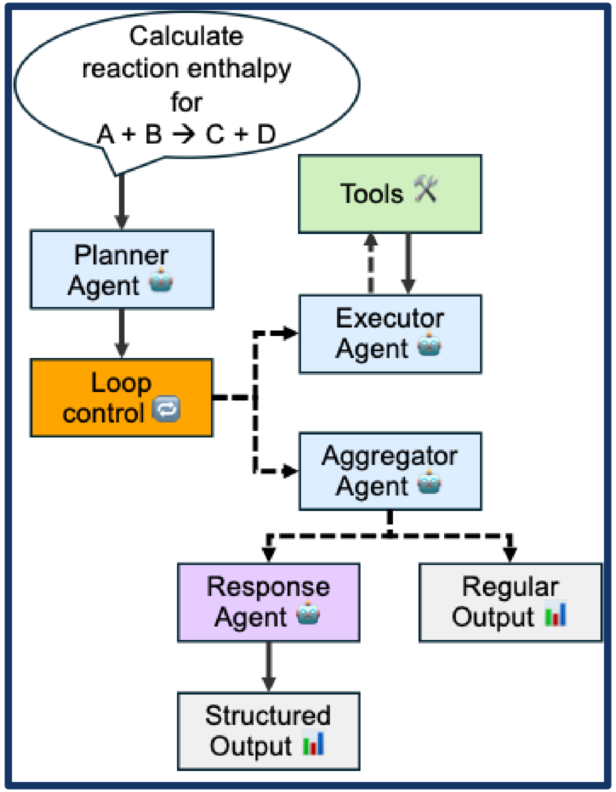
- Natural language interface: consistent interfaces to simulators, databases, instruments
- Automated planning: expand natural-language goals into workflows
- Smart parameter search & active learning to reduce experiments/simulations
- Failure recovery: read logs, adjust parameters, and retry safely
- Provenance capture: structured summaries, metadata, and reports
- Human-in-the-loop decisions at critical gates

Agentic Framework Landscape

- **LangGraph** (LangChain): graph-based control for stateful, tool-using agents
- **AutoGen** (Microsoft): multi-agent conversations with custom tools and human-in-the-loop
- **CrewAI**: role-based multi-agent teams with task decomposition and tools
- **LlamaIndex Agents**: retrieval-centric agents with tool orchestration
- **smolagents** (Hugging Face) — Minimalistic, reproducible agent runtime with tool calling and code execution. https://github.com/huggingface/smolagents
- **SuperAGI** — Open-source autonomous agents with dashboards, tools, and vector memory integrations.
- **AgentScope** (ByteDance) — Multi-agent framework emphasizing tool use and collaboration.

ChemGraph Design





Software suite

ChemGraph: An Agentic Framework for Computational Chemistry Workflows. (2025). Pham, T. D.; Tanikanti, A.; Keçeli, M. Under review.
Source code: https://github.com/argonne-lcf/ChemGraph

Hands-on time

- ChemGraph is open-source with an Apache 2.0 license: https://github.com/argonne-lcf/ChemGraph Contributions are welcome.
- You will be building a simple ReAct agent and then multi-agents with LangGraph demonstrating a mini app for ChemGraph.
- Homework: Implement your own multi-agent with your own tools and demonstrate with ALCF endpoints. Submit two files: <your_name>_tools.py and <your_name>_multi_agent.py and optionally requirements.txt. Make sure you make use of ALCF endpoints and your code runs with python <your_name>_multi_agent.py. Any additional dependency required should be listed in requirements.txt and must be installable by pip.